Área Departamental de Engenharia de Eletrónica e Telecomunicações e de Computadores

DRAG

Autores: [43586] Miguel Paixão
[43528] Diogo Martins
[43755] Luís Guerra

Relatório para a Unidade Curricular de Programação de Dispositivos Móveis da Licenciatura em Engenharia Informática e de Computadores

Professor: João Trindade

18-01-2021

Estrutura da nossa Aplicação DRAG

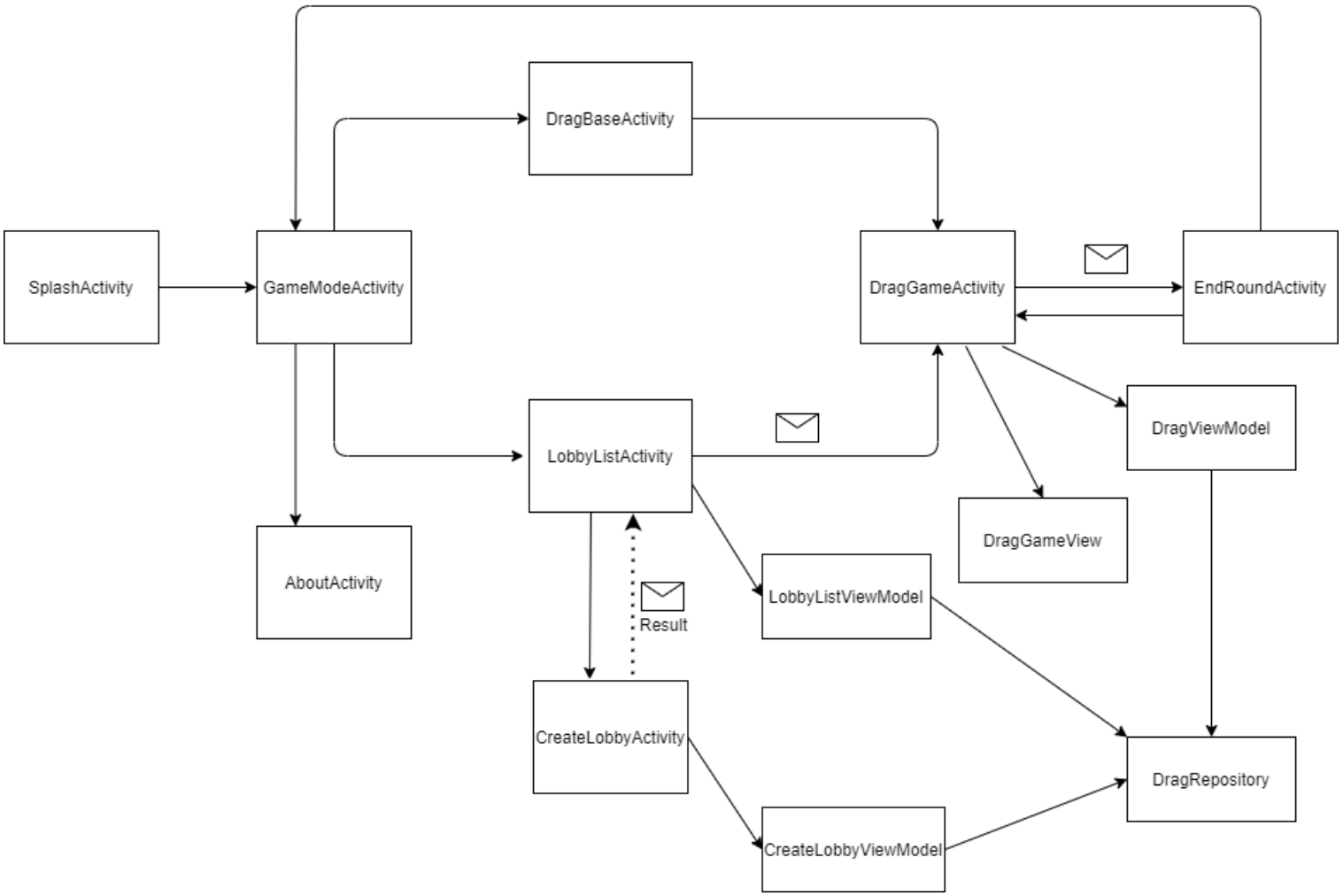

Figura 1 – Estrutura da aplicação Drag

Inicialmente quando o utilizador abre a aplicação é iniciada a atividade ***SplashActivity***.

SplashActivity

Esta atividade é responsável por mostrar o logo da aplicação. Se o utilizador não tocar no ecrã durante trinta segundos ou clicar no ecrã é redirecionado para a atividade ***GameModeActivity***.

GameModeActivity

Esta atividade é responsável por dar a escolha ao utilizador de se quer jogar em modo online ou local. Tem também o botão de ***AboutActivity*** que apresenta informação sobre os criadores.

Caso o utilizador escolha o modo local é redirecionado para a atividade ***DragBaseActivity***.

Caso o utilizador escolha o modo online é redirecionado para a atividade ***LobbyListActivity.***

DragBaseActivity

Esta atividade é responsável por obter as condições do jogo, ou seja, quantos jogadores vão jogar (mínimo cinco jogadores) e quantas rondas é que vão ser jogadas (mínimo de uma ronda).

Quando o utilizador clica no botão para começar é remetido para a atividade ***DragGameActivity***.

LobbyListActivity

Esta é responsável por listar todas as salas de jogo disponíveis para um jogador entrar. Esta atividade tem também dois botões, um para fazer atualização da lista de salas disponíveis e o outro para criar uma sala de jogo.

Para representar a lista de salas de jogos disponíveis foi preciso criar um *LiveData.* Este armazena uma lista de informação de todas as salas e sobre ela aplica um *observe,* quando houver uma alteração sobre esse *LiveData* a atividade faz a representação de cada uma das salas.

Para realizar a atualização da lista de salas, criou-se um botão, que sempre que é clicado obtém as salas de jogo presente na coleção de lobbies na *firestore*, faz a tradução para uma lista de informação de salas e atualiza o *LiveData*.

Sobre cada uma das salas, sempre que um utilizador clica numa sala, é chamado um *callback* que apresenta um *AlertDialog* ao utilizador se quer aceitar o desafio e qual é o seu nome. Se o utilizador aceitar é atualizada a informação da sala na *firestore*. Após aceite o desafio o utilizador é remetido para a ***DragGameActivity*** enviando a informação da sala e o jogado local.

No evento de *OnActivityResult,* ou seja, quando há um resultado, neste caso enviado no ***CreateLobbyActivity*** para a ***LobbyListActivity***, usando esse resultado é reencaminhada a informação da sala para a atividade ***DragGameActivity*** e também o jogador local.

CreateLobbyActivity

Esta atividade vai criar uma sala com o nome do utilizador que a criou e novamente com o número de jogadores e o número de rondas introduzidas pelo utilizador. Ao clicar para criar é publicado no *firestore*

um novo jogo. Após a publicação do jogo é enviado sobre a forma de resultado a informação da sala de jogo para a atividade ***LobbyListActivity***.

DragGameActivity

Esta atividade é responsável diretamente pelo "*flow*" do jogo. Inicia o jogo indo buscar palavras aleatórias através de um pedido (seja localmente ou à *api*), altera a apresentação da tela de jogo que o utilizador vê conforme o estado do jogo tendo como hipóteses: não começado, "desenhar palavra", "adivinhar palavra" e "escolher palavra". Antes do jogo se poder iniciar podemos visualizar ecrãs de espera como "esperando por outros jogadores" (caso o jogo seja *online*) e "por favor espere" enquanto o programa espera pela resposta do pedido de palavras.

Esta *activity* define também se é permitido clicar nos botões, a sua visibilidade quais as funções a executar ao clicar nos mesmos.

Para que tudo flua como esperado esta *activity* depende de um *viewmodel* chamado de ***DragViewModel***. Este *viewmodel* contém todas as informações pertinentes do início e do decorrer do jogo como as palavras, jogadores e informação de *lobby*, interagindo diretamente com o repositório da aplicação. Esta classe tem dados fulcrais à aplicação em forma de *LiveData* tais como as palavras, estado de jogo e jogo. Este estado é alterado ao longo do decorrer do programa e é esta *activity* que define que código executar conforme o estado de jogo corrente.

EndRoundActivity

Esta atividade apresenta o final da ronda ou o final do jogo dependendo da informação que lhe foi passada da atividade ***DragGameActivity***. Para alem disso mostra através de uma ***RecyclerView*** todas as jogadas que foram feitas durante a ronda. Se for o final do jogo ele regressa à atividade ***GameModeActivity*** caso contrário incrementa a ronda e começa a atividade ***DragGameActivity*** até chegar ao final das rondas.